Algorithm 2: Masking routine for MLM

function *tokenize_for_mlm(batch)*

```
// 1) Build paired texts: word - desc [SEP] greek_word - greek_desc
tokenized_texts ← [ ];
foreach (w, d, gw, gd) in zip(batch[word], batch[description], batch[greek_word],
  batch[greek_description]) do
    full_text ← f"{w} - {d} [SEP] {gw} - {gd}";
    ids ← tokenizer.encode(full_text, add_special_tokens=True, max_length=512, truncation=True);
    append(tokenized_texts, ids);
// 2) Pad to max length in batch
input_ids_tensors ← [tensor(ids) for ids in tokenized_texts];
input_ids_padded ← pad_sequence(input_ids_tensors, batch_first=True,
  padding_value=tokenizer.pad_token_id);
// 3) Apply MLM masking (15%)
(input_ids_masked, labels) ← mask_tokens(input_ids_padded, tokenizer, mlm_probability=0.15);
// 4) Attention mask: 1 for non-pad
attention_mask ← (input_ids_masked ≠ tokenizer.pad_token_id).long();
return { "input_ids": input_ids_masked,;
"attention_mask": attention_mask,;
"labels": labels };
```

function *mask_tokens(inputs, tokenizer, mlm_probability=0.15)*

```
// 1) Labels start as a copy of inputs
labels ← clone(inputs);
// 2) Sample candidate positions with Bernoulli(mlm_probability)
probability_matrix ← full(shape(labels), mlm_probability);
masked_indices ← bernoulli(probability_matrix).bool();
// 3) Exclude special tokens: PAD, CLS, SEP
special_mask ← zeros_like(labels, dtype=bool);
foreach tok ∈ {pad_id, cls_id, sep_id} do
    special_mask ← special_mask ∨ (labels == tok);
masked_indices ← masked_indices & not(special_mask);
// 4) Non-masked positions set to −100 in labels (ignored by loss)
labels[not(masked_indices)] ← −100;
// 5) Replace 80% of masked positions with [MASK]
replace_prob ← full(shape(labels), 0.8);
indices_replaced ← bernoulli(replace_prob).bool() & masked_indices;
inputs[indices_replaced] ← tokenizer.mask_token_id;
return (inputs, labels);
```